SMART CLOUD DATA MANAGEMENT SYSTEM USING ARTIFICIAL LEARNING

Sanjeev Prakashrao Kaulgud[1], Narender Chinthamu[2], Md Ziaur Rahman[3], Penumathsa Suresh Varma[4], Priyanka Nandal[5], S Hasan Hussain[6], Manideep Karukuri[7]

Article History: Received: 12.12.2022 **Revised:** 29.01.2023 **Accepted:** 15.03.2023

Abstract

The cloud offers many IT resources such as low-cost and flexibility of products across the Internet. Cloud became the most demanding infrastructure as many cloud providers seek greater business outcomes and environment. Cloud became more difficult as a result. It amply demonstrates that the intelligent cloud's new beginning is currently taking place. Optimizing the cost-effective cloud service configuration and resource allocation adaptability are just a few drawbacks that the cloud is currently experiencing. We can say that using machine learning to boost cloud management intelligence is becoming more and more popular. This article focuses on the architecture of deep reinforcement learning-based intelligent cloud resource management. Cloud will work efficiently and automatically negotiate the best configuration, starting from complex cloud environments by using Deep Reinforcement Learning. At last, We provide a concrete example to illustrate the remarkable power of deep reinforcement learning in the intelligent cloud.

Keywords: Cloud Data Management; Deep Learning; Machine Learning; Intelligent System

[1]Assistant Professor, School of Computer Science and Engineering, Presidency University, Itgalpura, Yelahanka, Bengaluru-560064, Karnataka, India,
[2]MIT (Massachusetts Institute of Technology) CTO Candidate, Enterprise Architect
[3]Assistant Professor, Department of Computer Science and Engineering, Presidency University, Itgalpura, Rajankunte, Yelahanka, Bangalore-560064, Karnataka, India
[4]Professor, Department of Computer Science and Engineering, Adikavi Nannaya University, Rajamahendravaram-533296, Andhra Pradesh, India,
[5]Associate Professor, Department of computer science and engineering, Maharaja Surajmal Institute of Technology, C4 Janakpuri, Delhi,
[6]Associate Professor, School of CSE and IS,Presidency University, Bengaluru, India,
[7]University of Texas at Arlington, MSBA Graduate, DALLAS, TEXAS United States,

Email: [1]sanjeev.kaulgud@gmail.com, narender.chinthamu@gmail.com,
[3]mdziaurrahaman@presidencyuniversity.in, [4]sureshvarmap@gmail.com, [5]priyankanandal@msit.in,
[6]hasanhussainme@gmail.com, [7]manideepkarukoori@gmail.com

DOI: 10.31838/ecb/2023.12.s3.074

1. Introduction

The reliability of applications now demands IoT sensors most widely. This made us develop a lot of smart technologies and control the equipment [1]. These are simple in terms of their cost parameters and size which makes them undergrowth for IoT applications. Due to the cost factor, this made IoT-based Applications use very easily and doesn't require a human management system for installation of it. IoT systems behave like they are engaged with vehicle driving modules because of their quick and accurate responses.[2-3].It has a task controller, wireless medium, and sensor with transmission module.No outside party can ever obstruct communication between the sensing element and the receiver because the proposed architecture uses single-ended Internet of Things transmission. To send the signal to an additional task controller, the transmission module is directly connected to the sensors.[4].Data collected from sensor devices is exchanged by the task controller, which consists of a microcontroller chip and IoT receiver module. Here, depending on the signal received from the IoT channel, the system's response will alter.

To enable the required action, the task controller immediately receives the sensor data collected by IoT sensors. Most applications won't ever store the obtained sensor data.[5].The medium is observed for a set period using those compromised and medical sensors to keep track of the measurement data records. Along with the IoT modules in these applications, external storage is provided to save the data.[6-7]. Another recent trend is the ability to store any kind of large amounts of data in the cloud for the required amount of time. Accessibility is the primary benefit of cloud data storage [8]. Anywhere in the world can easily access the measured values from IoT devices.

An edge computing model was implemented for making the cloud an easy process. Artificial intelligence techniques were incorporated into the architecture of edge computing to speed up and improve the computing process.[9]. The network simulator tool was used to analyze the performance of the edge computing technique, and the results show that edge computing performs better in terms of delay.[10]. To save elderly people in dire situations, a disaster management system was suggested. Numerous sensors were included in the system to monitor for floods, heavy rain, earthquakes, and other events. The IoT platform connects all sensors to let family members and the rescue team know when there is an emergency. With an ongoing power supply, the system operates independently and keeps going.[11]. A fog network has been used to create a network allocation system for speedier data transmission. This reduces the amount of energy needed for transmission while improving the speed of data transmission somewhat. A lot of data can be transmitted quickly thanks to the fog network.

2. Related works

In comparison to human and other manipulation processes, these data mining algorithms are quick and effective. By examining the sensor data with machine learning algorithms, a system for predicting river floods was proposed [12]. To determine the prediction accuracy, the work was validated using a random forest algorithm and LSTM.In contrast, the random forest model outperforms the LSTM model. To train and test the model as soon as possible, the data collected from the sensors is divided into various sections. The algorithm is then released for use in the standard analysis procedure [13]. To protect the transmitted IoT during testing, a process mining technique was developed. These algorithms can detect data misbehavior [14]. Attacks and data loss both have the potential to result in data loss and manipulation. To guarantee successful data reception, the process mining system monitors data transmission.[15].

During the cloud computing process, the likelihood of a data attack is decreased. A recommendation work on data mining algorithms was established to ascertain the most efficient algorithm for analyzing big data about their applications. According to the research, not every application will be compatible with every data mining algorithm [16]. For every application, a unique data mining algorithm needs to be developed to achieve exceptional system efficiency. The computational complexity, flexibility, and time required for the verification process must all be considered.

The cloud layer is located at the very top of the architecture. Cloud technology is the best in terms of processing and storage power [17]. To create the software services for citizens, all city data in this layer may be combined with other non-city data or other cloud data. One can carry out a variety of difficult tasks using the cloud platform that might not be possible at least patterns of the architectureD2C-DM in the city[18]. It is clear that a significant amount of IoT devices contribute to the system and generate enormous amounts of data in many smart city scenarios. Therefore, maintaining or storing the data close to the network's edge is essential to ensuring better facilities for any latency-sensitive services improving the effectiveness of a smart city approach overall.[19-20]. Notably, one of the fundamental requirements of any smart computing system is the efficient use of resources. It is necessary to preprocess this data before storing it and to remove any excess data to make efficient use of the system's overall storage resources.

In conclusion, cloud computing increases opportunities and capabilities for the efficient use of modern IT resources. With effective control algorithms, intelligent resource management in the cloud can maximize its advantages while demonstrating remarkable skill in automatic resource allocation and online.

3. Intelligent resource management-Architecture

Figure 1 illustrates the two main parts of an intelligent resource management architecture: IT resources, and an allocator, an intelligent resource manager, is made up of a controller, and a monitor; and an IT resource, is made up of big resource pools. To submit application requests with various demands, clients must first communicate with the controller. Based on application demands and knowledge of current resource utilization, the controller implements the algorithm from its resource schedule algorithm pool to satisfy application demands while taking system resource constraints into account. The intelligent resource management architecture's resource schedule algorithm pool, which is a key component, contains a variety of algorithms, including online algorithms and offline algorithms that combine both offline and online components [21]. The monitor is in charge of regularly providing the controller with information regarding system resource utilization and application quality of service, and the allocator is in charge of assigning applications to resource pools by the configuration chosen by the controllers. It collaborates with the allocator and monitors to intelligently distribute resources, the controller is a crucial part of a resource management architecture. The resource scheduler algorithm pool, which houses many control algorithms, is the brain of the controller.

The DRL algorithm described in this paper is an online algorithm that merges reinforcement learning and deep learning to directly generate, in a limited number of iterations from the raw application requirements, the ideal resource configuration, especially for high dimensional requirements. The controller, as shown in Figure 1, chooses a course of action based on the deep learning network, and as compensation, the application operating environment provides feedback data and a new environment state. After the deep neural network has been pretrained with RE, it is optimized using reinforcement learning experiences. As a result, the deep learning and reinforcement learning components can work in unison to make a way to unprocessed the demands of applications and intelligently determine a policy configuration in a set of limited steps.

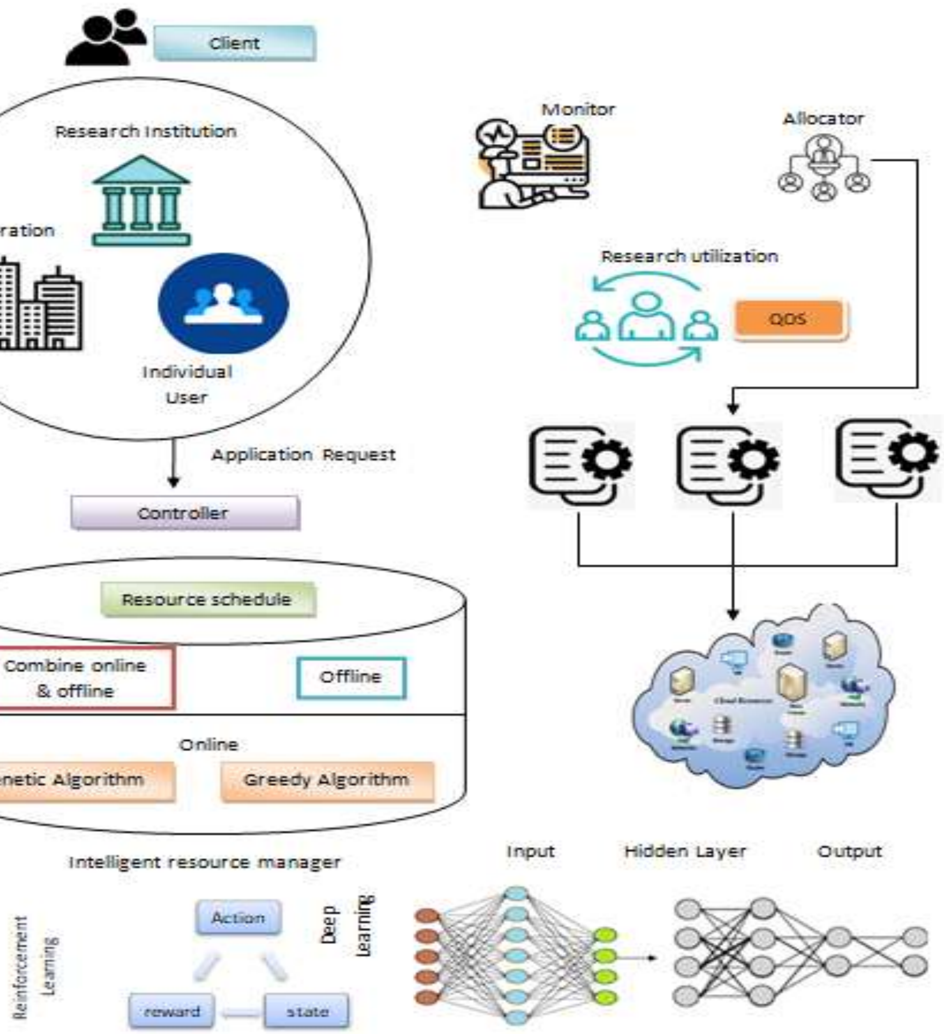

Figure 1: Intelligent resource management-Architecture

The resource pool is a fully managed cloud hosting choice with exceptional flexibility. A resource pool is an abstraction that users use to present and consume resources consistently. It stands for a collection of methods for resource management and classification used by service providers. Applications, hypervisors, virtual machines, and physical and virtual resources make up the typical five layers of a resource pool. After the allocator matches applications to the correct resource pools, an adequate number of resources are assigned for implementation

Deep Reinforcement Learning (DRL)

The agent can predict the ideal behavior from previous behavior thanks to a subfield of machine learning known as reinforcement learning. Q-learning, the most well-known and widely used reinforcement learning algorithm, bases its decisions on a Q-table when selecting actions. To hit the bricks arranged in a line at the top of the screen in the popular video game Breakout, for example, you control a movable paddle at the bottom of the screen by bouncing the ball. whenever a brick is struck, causing it to be destroyed and increasing your score, you are rewarded. The agent in Q-learning is only shown a small number of screen images, but it can choose a moving action based on prior experiences to increase the score. Past effects are typically stored in a Q-table along with Q-values, To evaluate the extended effects of a specific action in a given state. The Q-table will iteratively update based on the Bellman equation each time a new action is taken, and the agent will occasionally explore uncharted territory to avoid locally optimal solutions.

Deep learning has shown incredibly promising results in the identification of complex structures of high-dimensional data, such as speech, images, and space data, when a sufficient number of these transformations are combined. The neural network's offspring, deep learning, is a model-free function approximator that can easily handle the increasing computation and data loads. By combining traditional reinforcement learning with deep learning, The exceptional skill of DRL in feature extraction and function approximation is the reason it is being suggested.

4. Outcome and discussions

A Markov decision process is the most common way to formalize a reinforcement learning problem in DRL algorithms like the DQN model. Let's say the agent is located in a setting that is symbolized by a particular state. The agent can take certain actions in changing the environment into a different state, earn rewards, environment and then change the environment into a different state. An MDP is made up of a set of transition, states, and action rules, and, a,r,s9 are the experience of transition. A discrete series of states, deeds, and benefits are formed by one episode of this process:

$$\mathbf{s_0}, \mathbf{a_0}, \mathbf{r_0}, \mathbf{s_1}, \mathbf{a_1}, \mathbf{r_1}, \mathbf{s_2}, \dots \dots \mathbf{s_t}, \mathbf{a_t}, \mathbf{r_t}\mathbf{s_{t+1}}, \dots \dots ., \mathbf{s_{n-1}}, \mathbf{a_{n-1}}, \mathbf{r_{n-1}}, \mathbf{s_n} \quad (1)$$

Several measurement indices, including performance, durability, and cost efficiency are related to the goal of resource management. Here, to clearly explain the Replaced Encoder (RE) network (REN) algorithm, we set the goal of locating the best configuration policy while taking cost considerations into account. In REN, a state is a configuration of various resources, such as the quantity of CPUs and memories, based on the predefined target. A single resource can be adjusted by one unit, such as by increasing the CPU by one unit and is referred to as an action. Due to its complexity, we abandon the approach of representing rewards as a linear combination of performance and cost when they refer to a two-dimensional vector involving performance and cost. For example, the discount factor, which starts at 0.5 and enables the agent to balance the future and current prediction of rewards, can be adjusted based on system performance. Common fixed values are also often used as the starting point for other adjustable parameters. Just like DQN, formalization of MDP can be negotiated, which is due to Markov assumption ie., current state and action dependency shown in Figure 2. Using its selection policy and the transition rule, the agent can then choose an action to take and change states. Deep learning comes into play, after saving the finished transition sequence in a replay memory, The Q-function is estimated by REN using a nonlinear approximator, as opposed to the Bellman equation, which estimates and updates the Q-function individually. the corresponding Q-values are produced by the nonlinear approximator, which accepts states and actions as inputs. The Q-value in REN is a two-dimensional vector made up of performance and cost, which is precisely the normalized reward. The Q-value is used to evaluate the long-term effects of a specific action in a given state, and REN aims to learn the best configuration policy that satisfies the predefined performance with the lowest cost.

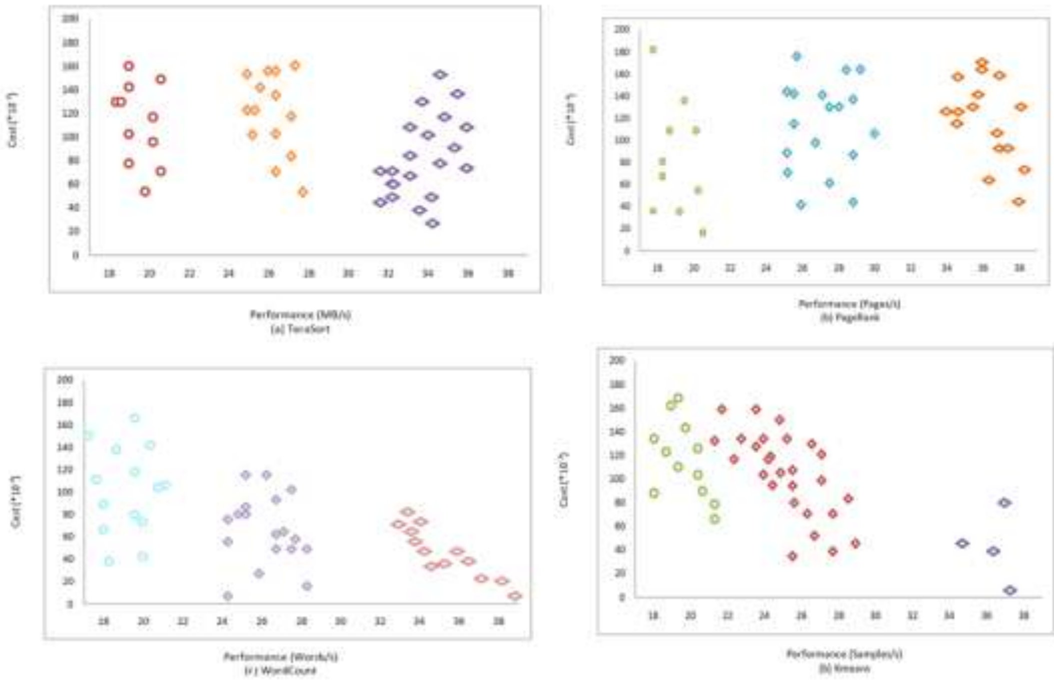

Figure 2: Measurements of the performance and cost

Stability issues are brought up by a reinforcement learning algorithm's use of a nonlinear function approximator. As a result of being sequential, the input data does not adhere to the independent, identical distribution that requires neural networks. For training the Q-network, we use a mechanism called experience replay, which saves transition sequences that have already happened. Thus, it not

only eliminates data correlation but also permits the network to draw lessons from all prior experiences. In addition, the data distribution will change significantly, and the policy may fluctuate with minor Q-value adjustments. The typical solution to this issue is to periodically update fixed network parameters while freezing the Q-network. The reward scale is the final problem, which when backpropagated will result in an unstable state for naive Q-learning gradients. This is fixed by normalizing the network to a reasonable range, limiting error derivations, and adapting clipping rewards.
The following is a description of the entire REN execution flow: The controller first discusses the requirements of the client application with the agent. The agent then reverts to its initial configuration, investigates its surroundings, and chooses a course of action based on what it knows at the moment. The agent can be forced to investigate every possibility initially at random before relying more heavily on its experience by we using an adaptive e-greedy exploration policy. The agent receives feedback and enters a new state based on the decision made regarding the transition rule and action. The agent stores a transition sequence it has completed in a replay memory so it can be used to train the Q-network. The agent chooses random transition sequences from the replay memory and trains the Q-network based on the loss function as part of the network training process. The agent enters the subsequent iteration based on the new state after training is finished and iterates until the optimization goal is achieved.

4.1 Efficiency Comparisons

We examine the average cost reduction of the best configuration policy to assess the proposed REN algorithm's accuracy, SmartYARN,15 and RENwhich uses fundamental reinforcement learning, a more recent approach to intelligent resource management, to arrange primarily for low-dimensional data and the best course of action, as follows:

$$\textbf{Cost reduction ratio}_{\textbf{optimal}} \leftarrow \frac{\textbf{cost}_{\textbf{real}}^{\textbf{max}} - \textbf{cost}_{\textbf{real}}^{\textbf{min}}}{\textbf{cost}_{\textbf{real}}^{\textbf{max}}} \quad (2)$$

$$\textbf{Cost reduction ratio}_{\textbf{SAQN}} \leftarrow \frac{\textbf{cost}_{\textbf{real}}^{\textbf{max}} - \textbf{cost}_{\textbf{SAQN}}^{\textbf{min}}}{\textbf{cost}_{\textbf{real}}^{\textbf{max}}} \quad (3)$$

$$\textbf{Cost reduction ratio}_{\textbf{SmartYarn}} \leftarrow \frac{\textbf{cost}_{\textbf{real}}^{\textbf{max}} - \textbf{cost}_{\textbf{SmartYarn}}^{\textbf{min}}}{\textbf{cost}_{\textbf{real}}^{\textbf{max}}} \quad (4)$$

The fact that the outcomes of the optimal policy and REN are generally identical or extremely similar in other instances demonstrates REN's remarkable capacity for figuring out the best way to allocate resources. Additionally, by utilizing every reduction result in Figure 3, REN can reduce the cost of the ideal configuration policy by an average of 99.21%, while SmartYARN can do so by 98.13%, demonstrating the superior capability of REN in handling low-dimensional data.

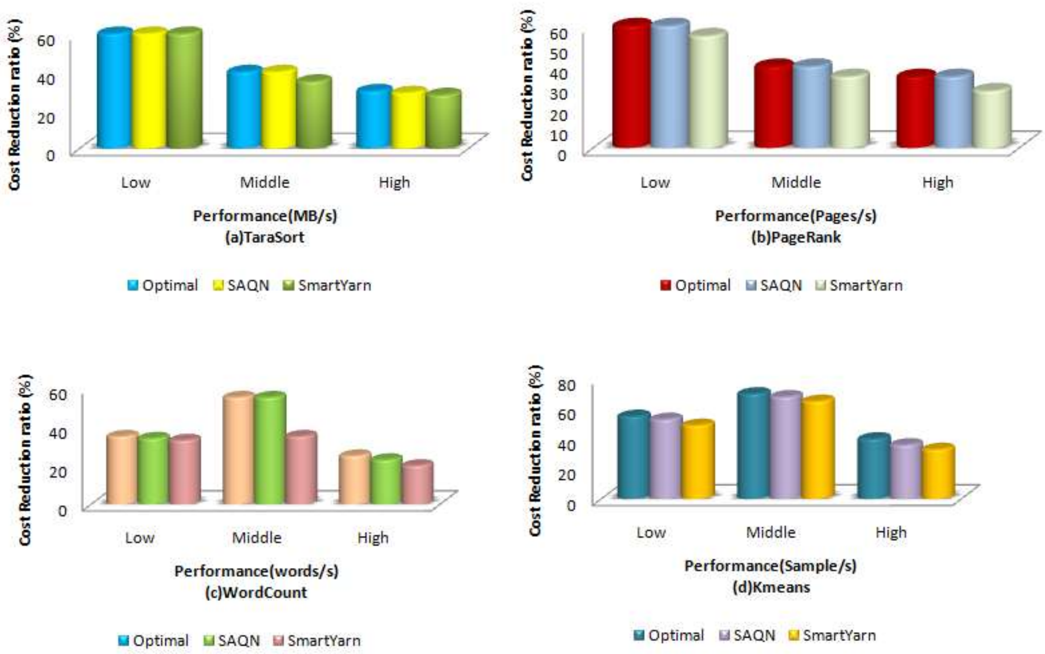

Figure 3: Optimal configuration in terms of Accuracy

To confirm the effectiveness of REN, we contrasted its iterations with those of Smart YARN, the exhaust algorithm, and REN. The exhaust algorithm can identify the ideal resource configuration by exploring every scenario. To achieve the best outcome, the exhaust algorithm requires 81 iterations. Figure 4 illustrates how Smart YARN requires an average of 15.89 iterations while REN only requires an average of 8.07 iterations to learn the best configuration policy, demonstrating REN's superiority in online deployment.

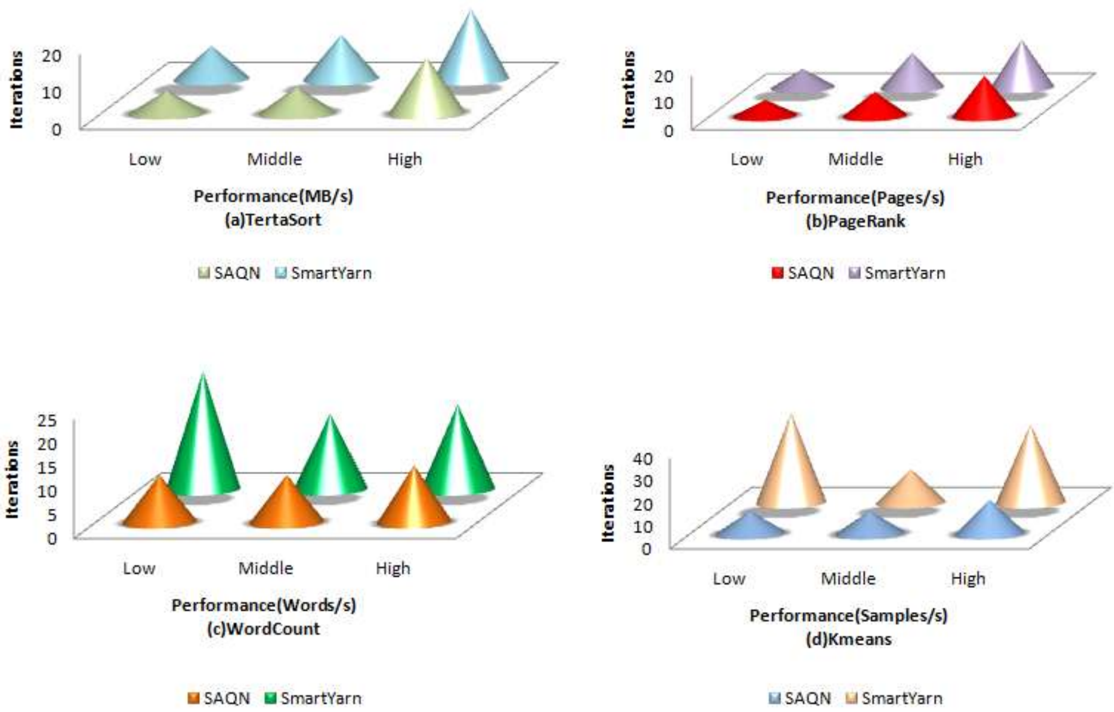

Figure 4: Efficiency comparisons between REN and Smart YARN.

5. Conclusions

Through a finite number of learning processes—on average, 8.07 iterations—REN can always deliver the ideal configuration policy that satisfies the specified requirements. DRL is a useful method of resource management because its primary cost is the time spent looking for the best course of action*Because it optimizes the action-value function using real-time measurements when compared to offline algorithms, REN demonstrates its extraordinary capacity to negotiate a better configuration policy. When compared to other online algorithms, REN also demonstrates especially for applications that must run for several hours or even several days, its benefit in the reduction of iterations.REN is typically a better option in resource management with multiple optimization goals because it combines reinforcement learning, which derives the optimal policy directly from the raw application inputs, and deep learning, which accelerates the learning process, especially for complex and high-dimensional data.

6. References

Villegas-Ch, W., Arias-Navarrete, A., & Palacios-Pacheco, X. (2020). Proposal of an Architecture for the Integration of a Chatbot with Artificial Intelligence in a Smart Campus for the Improvement of Learning. Sustainability, 12(4), 1500.

Stergiou, C. L., &Psannis, K. E. (2022). Digital Twin Intelligent System for Industrial IoT-based Big Data Management and Analysis in Cloud. Virtual Reality & Intelligent Hardware, 4(4), 279-291.

Saqlain, M., Piao, M., Shim, Y., & Lee, J. Y. (2019). The framework of an IoT-based industrial data management for smart manufacturing. Journal of Sensor and Actuator Networks, 8(2), 25.

Ahsan, U., &Bais, A. (2017, April). Distributed big data management in smart grid. In 2017 26th Wireless and Optical Communication Conference (WOCC) (pp. 1-6). IEEE.

Hamid, R. A., Albahri, A. S., Alwan, J. K., Al-Qaysi, Z. T., Albahri, O. S., Zaidan, A. A., ... &Zaidan, B. B. (2021). How smart is e-tourism? A systematic review of smart tourism recommendation system applying data management. Computer Science Review, 39, 100337.

Poorvadevi, R., Kowsigan, M., Balasubramanie, P., &Rajeshkumar, J. (2020). IoT-Based Secure Smart Infrastructure Data Management. Role of Edge Analytics in Sustainable Smart City

Development: Challenges and Solutions, 243-256.

Blessy, J. A. (2021, February). Smart irrigation system techniques using artificial intelligence and iot. In 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV) (pp. 1355-1359). IEEE.

Debauche, O., Mahmoudi, S., Manneback, P., &Lebeau, F. (2021). Cloud and distributed architectures for data management in agriculture 4.0: Review and future trends. Journal of King Saud University-Computer and Information Sciences.

Kamruzzaman, M. M. (2020, July). The architecture of smart health care system using artificial intelligence. In 2020 IEEE international conference on multimedia & expo workshops (ICMEW) (pp. 1-6). IEEE.

Rehman, A., Haseeb, K., Saba, T., &Kolivand, H. (2021). M-SMDM: a model of security measures using the Green Internet of Things with cloud-integrated data management for smart cities. Environmental Technology & Innovation, 24, 101802.

Kristiani, E., Yang, C. T., Huang, C. Y., Ko, P. C., &Fathoni, H. (2020). On the construction of sensors, edge, and cloud (ISEC) framework for smart system integration and applications. IEEE Internet of Things Journal, 8(1), 309-319.

Abbasi, M. A., Memon, Z. A., Syed, T. Q., Memon, J., &Alshboul, R. (2017). Addressing the future data management challenges in iot: A proposed framework. International Journal of Advanced Computer Science and Applications, 8(5).

Ezhilarasi, T. P., Dilip, G., Latchoumi, T. P., & Balamurugan, K. (2020). UIP—a smart web application to manage network environments. In Proceedings of the Third International Conference on Computational Intelligence and Informatics (pp. 97-108). Springer, Singapore.

Sheng, T. J., Islam, M. S., Misran, N., Baharuddin, M. H., Arshad, H., Islam, M. R., ... & Islam, M. T. (2020). An internet of things-based smart waste management system using LoRa and tensorflow deep learning model. IEEE Access, 8, 148793-148811.

Piccialli, F., Di Somma, V., Giampaolo, F., Cuomo, S., &Fortino, G. (2021). A survey on deep learning in medicine: Why, how, and when?. Information Fusion, 66, 111-137.

Tiwari, P., Uprety, S., Dehdashti, S., & Hossain, M. S. (2020). TermInformer: unsupervised term mining and analysis in biomedical literature. Neural Computing and Applications, 1-14.

Latchoumi, T. P., Ezhilarasi, T. P., & Balamurugan, K. (2019). Bio-inspired weighed quantum particle swarm optimization and smooth support vector machine ensembles for identification of abnormalities in medical data. SN Applied Sciences, 1(10), 1-10.

Smith, A. (2020). Shifting towards Digital Epidemiological Surveillance, Virtualized Care, and Smart Internet of Things-enabled Mobile-based Health Monitoring Systems in Response to COVID-19. American Journal of Medical Research, 7(2), 36-43.

Ahmed, I., Jeon, G., &Piccialli, F. (2021). A deep-learning-based smart healthcare system for patient’s discomfort detection at the edge of Internet of things. IEEE Internet of Things Journal, 8(13), 10318-10326.

Allgaier, J., Schlee, W., Langguth, B., Probst, T., &Pryss, R. (2021). Predicting the gender of individuals with tinnitus based on daily life data of the TrackYourTinnitus mHealth platform. Scientific Reports, 11(1), 1-14.

Mendo, I. R., Marques, G., de la Torre Díez, I., López-Coronado, M., & Martín-Rodríguez, F. (2021). Machine learning in medical emergencies: a systematic review and analysis. Journal of Medical Systems, 45(10), 1-16.